



Original citation:

Alexander-Craig, I. D. (1987) CASSANDRA-II : a distributed Blackboard system. University of Warwick. Department of Computer Science. (Department of Computer Science Research Report). (Unpublished) CS-RR-090

Permanent WRAP url:

<http://wrap.warwick.ac.uk/60786>

Copyright and reuse:

The Warwick Research Archive Portal (WRAP) makes this work by researchers of the University of Warwick available open access under the following conditions. Copyright © and all moral rights to the version of the paper presented here belong to the individual author(s) and/or other copyright owners. To the extent reasonable and practicable the material made available in WRAP has been checked for eligibility before being made available.

Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.

A note on versions:

The version presented in WRAP is the published version or, version of record, and may be cited as it appears here. For more information, please contact the WRAP Team at: publications@warwick.ac.uk



<http://wrap.warwick.ac.uk/>

Research report 90

CASSANDRA-II: A DISTRIBUTED BLACKBOARD SYSTEM

Iain D Craig

(RR90)

Abstract

The blackboard problem solving architecture has been used in a number of real-time and near real-time domains with some success. The architecture is usually employed when control becomes complex. The control component of a blackboard system is centralised and combines both local and global control. In this paper, a system is described which distributes control across the blackboard. The system, CASSANDRA-II, has been used in the controlled airspace monitoring domain. CASSANDRA-II is a distributed system in which the usual abstraction levels of the blackboard are replaced by level managers. Level managers contain schedulers which make control decisions only for their own level manager. Global control is achieved by a separate scheduler.

Department of Computer Science
University of Warwick
Coventry
CV4 7AL, UK

Jan 1987

1. INTRODUCTION

The blackboard architecture (Ertan, 1975; Hayes-Roth, 1983) has been used for a variety of real time or near real time problems. It has also been used in distributed problem solving experiments (Lesser, 1983). It has been plausibly argued that the blackboard architecture might be a general model of problem solving (Hayes-Roth, 1983).

Despite considerable interest in the architecture, and its successful application in a number of domains, it suffers from a number of problems, particularly where control and organisation are concerned. The core of a blackboard system is a global database (the blackboard) which is organised as an abstraction hierarchy: this hierarchy serves as an outline plan for the problem solving process (Feigenbaum, 1978). Not all problem domains can easily be structured according to a strict hierarchy (e.g., comments in (Hayes-Roth, 1979)). In some cases, a somewhat unnatural and highly abstract hierarchy must be invented in order to fit the architecture. Coupled to the abstraction problem is the micro-level problem: abstraction levels frequently gloss over many abstractions which fall within a given level (these finer-grained abstractions are called micro-levels in (Lesser, 1977)). These problems suggest that the imposition of hierarchy may be unwarranted in some cases and, thereby, complicate the architecture.

The second major problem with the blackboard architecture is scheduling. Blackboard systems are frequently used to solve problems which require complex control regimes. As Balzer et al. note (Balzer, 1980), scheduling is the most difficult aspect of building a blackboard application. In the blackboard architecture, all decisions are made by a central scheduler: local and global control are mixed. Hayes-Roth (Hayes-Roth, 1984) has developed a theory of control based upon planning. The Hayes-Roth theory requires explicit representation of control decisions within the system.

In this paper, a system called CASSANDRA-II (Craig, 1986) is described. It was designed to monitor air traffic in controlled airspace in approximately real time. CASSANDRA-II is based on the blackboard architecture, but the requirements of reliability and graceful degradation suggested changes to the traditional architecture. The major changes concern control and abstraction. The system contains fewer abstraction hierarchies than usual; it also makes a distinction between local and global control, a distinction which makes the construction of schedulers somewhat easier than is usually the case.

In the next section, the application domain is briefly outlined. In section three, the CASSANDRA-II system is described. Section four discusses the approach to control in the system. Section five is concerned with reliability and the system's contribution to distributed problem solving.

Section six summarises the main points.

2. TASK DOMAIN

CASSANDRA-II is concerned with the monitoring of aircraft movements in controlled airspace. The system warns the user of any violations (conflicts or conflictions) of air traffic control rules. Aircraft are controlled relative to a set of rules, some of which apply to specific kinds of airspace (sector rules), some of which apply to any controlled airspace. One aspect of the latter rules is separation: aircraft, for safety and efficiency reasons, must be separated in both the horizontal and the vertical. There are, in addition, rules relating to the altitude at which aircraft must fly.

The system contains knowledge sources to perform vertical, lateral and longitudinal separation checks and to check that each aircraft is flying at the correct altitude for its heading. When any of these checks fails, a message is sent to the user stating the type of violation, the call-sign of the aircraft and other useful information.

In addition, CASSANDRA-II contains knowledge sources to perform other checks. These knowledge sources remove from consideration all aircraft which are not inside the sector being monitored (radar will produce returns for all visible aircraft, not just those within the sector) and remove aircraft which are too far separated to engage in any conflict. One knowledge source is responsible for detecting crashes between aircraft. If aircraft have crashed, the user should be informed immediately in order to direct emergency services to the site.

The task domain is quite interesting structurally. It contains completely separated types of knowledge: there is no interaction between separation types and separation does not interact with cruise level allocation. The only requirement is that an aircraft outside the monitored airspace should be removed from consideration and that crashes be detected before any other checks are applied. Beyond these requirements, the separation and cruise level checks may be performed in parallel. These observations led to the development of the CASSANDRA-II architecture.

3. SYSTEM STRUCTURE

CASSANDRA-II is structured as a set of nine inter-communicating level managers. Communication between level managers is always uni-directional. Level managers correspond to abstraction levels in more conventional blackboard systems. There are no hierarchical relationships between level managers although the separation and cruise level allocation processes are associated with two level managers, one of which is responsible for pruning away

aircraft which cannot possibly engage in conflicts, the other of which performs the checks. The relationship between the level managers in each of these pairs is far closer to producer-consumer than it is to abstraction. Each level manager in the system operates independently of all others: the only exception to this is when a level manager has no more information to process and waits for input from another level manager or the surveillance radar system[1].

The gross structure of the system is shown in fig. 1.

The core of system is the level manager concept. Level managers are designed to be a device for collecting related knowledge items (blackboard entries) and relevant knowledge sources together. Level managers contain a database (called the level by analogy with the blackboard model) which holds entries or hypotheses. The entries in the level are used to trigger the local knowledge sources. The knowledge sources held within a level manager represent knowledge and expertise which is purely local to the level manager: a knowledge source instance belongs to one and only one level manager. A knowledge source may add, update, create and delete entries in the level manager to which it belongs; knowledge sources can also post entries to other level managers by sending messages (knowledge sources cannot directly side-effect levels other than those to which they belong).

The intention of dividing the system into level managers is to permit largely autonomous components, each of which has a scheduler tailored to its needs. Each level manager contains a scheduler and scheduling database. The scheduler is concerned with purely local control decisions -- no global control decisions are made by a level manager. In addition to a local scheduler and its database, each level manager contains a knowledge source precondition and condition matcher and a knowledge source action interpreter: these permit different representations in different level managers.

The provision of these components entails that each level manager is completely independent of all others. Level managers communicate with each another by passing messages. The result is that CASSANDRA-II exhibits greater modularity than a more traditional blackboard system. In traditional blackboard systems, the primary source of modularity is the knowledge source (each KS is ignorant of the existence of all others -- this also applies to CASSANDRA-II): in the present system, level managers can be added or removed with very little difficulty.

[1] In the current implementation of the system, radar input is replaced by input from a simulator.

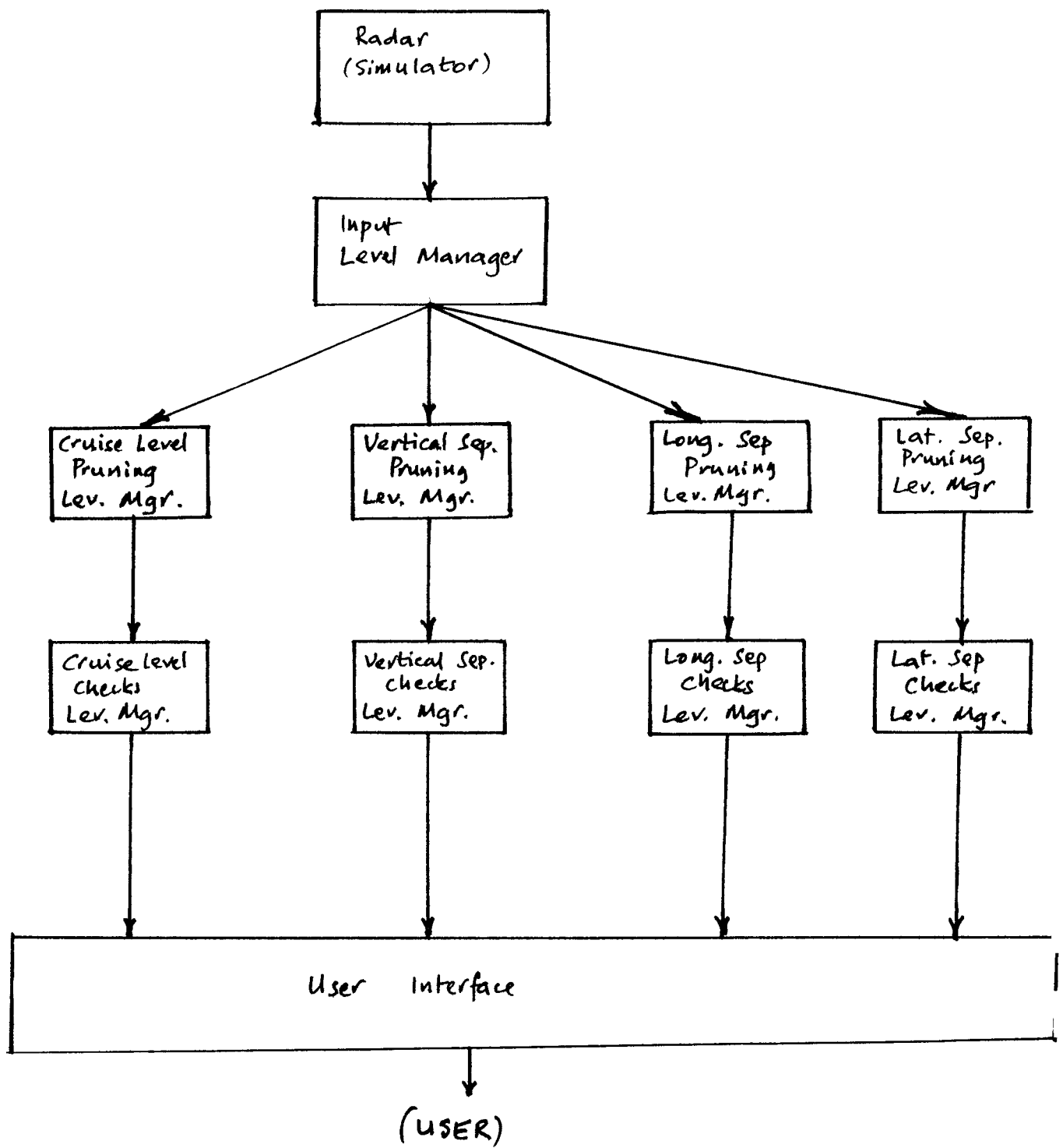


fig. 1

The structure of a level manager is shown in fig. 2.

4. CONTROL ISSUES

In conventional blackboard systems, control is centralised in a global scheduler. The global scheduler is responsible for selecting knowledge source activations from a set of candidates and for focussing the system's attention on various regions of the blackboard. Scheduling has been regarded for a long time as one of the more difficult aspects of the blackboard architecture. Scheduler designs have frequently been very complex (Hayes-Roth, 1977). The centralised scheduler is also concerned with both local (which KS to execute within a region of the blackboard) and global (which region of the blackboard to concentrate on next) control problems.

In CASSANDRA-II, a different approach was adopted. The analysis of the problem domain showed that different parts (the level managers) could operate relatively independently of each other: this suggested that control could be localised within level managers to a large extent. The second issue was that there are different control strategies which apply to the airspace monitoring domain: some level managers effect their control using distance information, but the cruise level allocation manager does not have this kind of information available. In addition, the interface to radar (the input level manager) and some of the pruning level managers require sequences of knowledge source executions. These requirements could be met by a centralised scheduler, but the final requirement that there be, at least, the possibility of distribution over some communication network strongly indicated that control should not be centralised.

These observations are supported in the current version of the system. Each level manager has a local scheduler. The local scheduler is charged with the control of only the level manager in which it resides. Global scheduling decisions cannot be made by a local scheduler. Local schedulers can, however, send messages to other level managers in order to request some form of behaviour.

In the controlled airspace monitoring domain, an example of using message passing to effect non-local control is as follows. If an aircraft has been detected to be on an incorrect flight level given its heading, it is possible that it will engage in conflicts with other aircraft. Messages can be sent to the separation level managers (in particular, the vertical separation manager -- vertical separation conflicts will be the most probable) so that attention can be focussed on any events concerning that particular aircraft.

In a more conventional blackboard system, the decision to focus on the offending aircraft would be taken by the

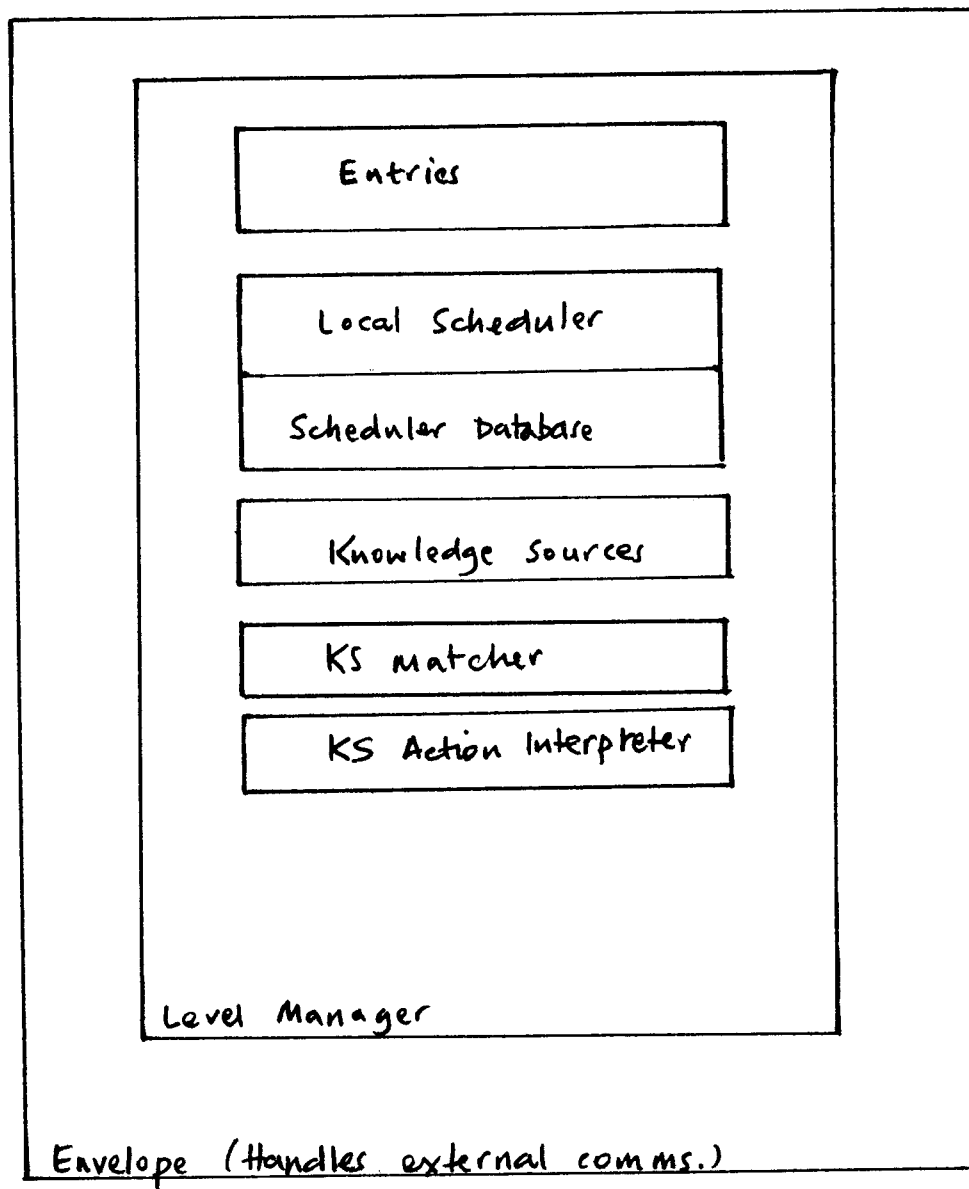


fig. 2

central scheduler. That suggests that the scheduler must monitor for violations (or have representations of violations) or must be aware of the intentions of the tasks it selects for execution. In CASSANDRA-II, the methodology which emerged was one of doing what is necessary where it is necessary: for control, this entailed collecting control decisions about a particular facet of the airspace into one place - the level manager responsible for that facet.

The methodology outlined above has a number of effects. The first effect is that the schedulers associated with each level manager are quite simple. They are concerned only with the scheduling of tasks within the level manager and are totally independent of any global focus decisions. Where focus decisions have to be made, a global scheduler is used. The system operates on a dual-scheduling system. This second point entails that the global scheduler makes decisions which determine how the level managers as individuals are to operate. The division between local and global control introduces a second benefit: the design of the system's scheduling mechanisms can be divided into two complementary components. The result of this division is that the two types of control component become smaller than usual and, because local schedulers are simpler, they tend to be faster than the comparable monolithic scheduler[2].

In the current implementation of the system, the global scheduler is concerned only with selecting the next level manager to activate. The aim of the present system is to perform as fast as possible. A version of the system without a global scheduler is under construction. The new version is to be distributed across the Departmental network. In the distributed version, the same methodology will be applied. In this experiment, schedulers will be concerned with what they are good at and will not be permitted to direct, in any way, the actions of other level managers, but will only be able to make suggestions. This decision is partly motivated by the fact that the global scheduler in the current version of the system is responsible for serialising level manager activations on a uni-processor. It is expected that some component will be required to monitor control messages from other level managers: this entails an extension to the level manager in which another, interacting, but logically separate scheduler will be provided.

5. DISTRIBUTION AND RELIABILITY

CASSANDRA-II was designed to allow distribution over many processors. As has been seen, this fact impacted on

[2] CASSANDRA-II's predecessor, CASSANDRA-I had a centralised scheduler. An experiment was performed to compare the relative performances: CASSANDRA-II performed better in terms of response time and accuracy of detection.

many of the design decisions, particularly where control is concerned.

It is possible to distribute the system in a variety of ways. One way to perform distribution is to allocate a separate processor to each level manager. This method of distribution was the one we had in mind when designing the system. By allocating a separate processor to each level manager, there is a natural mapping between the system's architecture and conventional hardware.

The allocation of a level manager to a single processor has reliability implications. In air traffic control, it is essential for systems to be as reliable as possible. If one assumes that there are more processors than level managers or else that a level manager does not consume all the resources of each processor in the network, it becomes possible for level managers to migrate from one processor to another in a fairly obvious fashion. This kind of migration might occur when there is either a hardware or a software failure.

When migration occurs, a new instance of the failed level manager is started on the new processor: the new level manager instance begins processing from scratch (that is, no attempt is made to recover any of the information stored in the failed level manager). In practise, this causes no problems when the time required to migrate and start a level manager on a new processor is less than the radar sweep time (on average, six seconds).

The facts that the system has been developed on a small uni-processor and that its performance is adequate suggest that the kind of distribution described here should be possible for nodes in a communication network with only modest power.

6. CONCLUSIONS

CASSANDRA-II, a variant of the blackboard architecture, has been described. It is a system designed for high reliability in a distributed, real time problem solving environment. The architecture is highly modular and lends itself naturally to parallel evaluation.

Control within the system is based upon the principle of controlling knowledge source activation in the place where it is required. The architecture of CASSANDRA-II provides a control mechanism in each system component. Knowledge sources are scheduled on a purely local basis. This entails that each scheduler can be more easily tailored to the task in hand. There is a methodological corollary to this: it is required that control be divided into local and global components, each of which is far simpler to design and construct than the analogue in a more traditional blackboard system.

The system was designed to be as simple as possible. This approach was adopted partly for reliability reasons (it is easier to maintain simple systems) and partly because of runtime considerations. Many of the data structures are fixed length which suggests that re-coding in ADA or C would be easily possible in order to extract better runtime performance.

A rough performance evaluation has been done. On a uni-processor system (a VAX 11/750 under 4.2BSD UNIX), CASSANDRA-II is able to handle between 2.2 and 3 times the current daily load at the London Air Traffic Control Centre, West Drayton, Middlesex, England. From these results, we conclude that the experiment was a success.

REFERENCES

- [Balzer, 1980] Balzer, R., Erman, L., London, P. and Williams, C. HEARSAY-III: A Domain Independent Framework for Expert Systems. Proc. First Annual Conference on Artificial Intelligence, pp. 108 - 110, 1980
- [Craig, 1986] Craig, I.D., Decentralised Control in A Blackboard System, Ph.D. Thesis, Department of Computing, University of Lancaster, Lancaster, UK
- [Erman, 1975] Erman, L.D. and Lesser, V.R., A Multi-level Organisation for Problem Solving Using Many, Diverse, Cooperating Sources of Knowledge. Proc. IJCAI 4, Vol. 2, pp. 483 - 490, 1975
- [Feigenbaum, 1978] Feigenbaum, E. and Nii, H.P., Rule-based Understanding of Signals. Pattern -Directed Inference Systems, D.A. Waterman and F. Hayes-Roth (Eds.), Academic Press, 1978
- [Hayes-Roth, 1977] Hayes-Roth, F. and Lesser, V.R., Focus of Attention in the Hearsay-II speech understanding system Proc. IJCAI 5, pp. 27 - 35, 1977
- [Hayes-Roth, 1979] Hayes-Roth, B. and Hayes-Roth, F., A Cognitive Model of Planning, Cognitive Science, Vol. 3., pp. 275 - 310, 1979

[Hayes-Roth, 1983] Hayes-Roth, B. The Blackboard Architecture:
A General Framework for Problem Solving?
Report No. HPP-83-30, Heuristic Programming
Project, Computer Science Dept.,
Stanford University, Palo Alto, CA, May 1983

[Hayes-Roth, 1984] Hayes-Roth, B. A Blackboard Model of Control.
Report No. HPP-83-38, Heuristic Programming Project,
Computer Science Department,
Stanford University, Palo Alto, CA.,
June 1983 (Revised December, 1984)

[Lesser, 1977] Lesser, V. R. and Erman, L.D.,
A Retrospective View of the Hearsay-II Architecture,
Proc. 5th IJCAI, Cambridge, Mass., pp. 790 - 800,
1977

[Lesser, 1983] Lesser, V., Corkhill, D.,
The Distributed Vehicle Monitoring Testbed:
a tool for investigating distributed problem
solving networks,
AI Magazine, Vol. 3, no. 4, pp. 15 - 33,
Fall, 1983